BRIDGING THE ARCHIPELAGO BETWEEN ROW-STORES AND COLUMN-STORES FOR HYBRID WORKLOADS

Joy Arulraj, Prashanth Menon, and Andy Pavlo
Carnegie Mellon University

SIGMOD 2016
REAL-TIME ANALYTICS

Transaction DBMS

Data Migration

Analytics DBMS
## TRANSACTIONAL VS ANALYTICS DBMS

<table>
<thead>
<tr>
<th></th>
<th>TRANSACTIONAL DBMS</th>
<th>ANALYTICS DBMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Layout</td>
<td>Row-store</td>
<td>Column-store</td>
</tr>
<tr>
<td>Query Processing</td>
<td>Tuple-at-a-time</td>
<td>Attribute-at-a-time</td>
</tr>
<tr>
<td>Write-Intensive Workloads</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>Read-Intensive Workloads</td>
<td>✗</td>
<td>✔</td>
</tr>
</tbody>
</table>
LIMITATIONS OF BIFURCATED SETUP

✗ Time taken to transfer data to analytics DBMS
✗ Overhead of maintaining two separate DBMSs

How can we do real time analytics on fresh data with lower overhead?
HYBRID DATABASE SYSTEM

- Hybrid storage layouts based on workload
  - Prior work: Data Morphing, HYRISE, HANA, H2O

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
<th>SALARY</th>
<th>CITY</th>
<th>STATE</th>
</tr>
</thead>
</table>

Hybrid-store for cold data

Row-store for hot data
Multiple execution engines for different layouts

Extra synchronization and query processing overheads

Can we design a single execution engine that can process tuples stored in different layouts?
KEY IDEA: LOGICAL TILE

✅ Abstract the hybrid layout from the execution engine
KEY IDEA: LOGICAL TILE

Abstract the hybrid layout from the execution engine
SELECTION & JOIN OPERATORS

### R

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>201</td>
<td>ITEM-1</td>
</tr>
<tr>
<td>2</td>
<td>202</td>
<td>ITEM-2</td>
</tr>
<tr>
<td>1</td>
<td>203</td>
<td>ITEM-3</td>
</tr>
</tbody>
</table>

### S

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>201</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>202</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>203</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>204</td>
<td>40</td>
</tr>
</tbody>
</table>

### R.A = 1

- {B, C}
  - 1
  - 3

### R JOIN S

- R.A = 1 & S.X = 2
- & R.B = S.Y

### R.B = S.Y

<table>
<thead>
<tr>
<th>{B, C}</th>
<th>{Y, Z}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

### Logical Tile
LAYOUT-AGNOSTIC EXECUTION ENGINE

- All operators produce and consume logical tiles
  - *Aggregation, Projection, etc.*

Logical tile algebra can be used to build a layout-agnostic engine for real-time analytics.
How should the DBMS adapt the hybrid storage layout with workload shifts?
KEY IDEA: DYNAMIC LAYOUT ADAPTATION

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Cold Tile Group

---

Time

Machine Learning Model
ADAPT Benchmark

- Dynamic hybrid workload
- Row-store, column-store, and hybrid-store
- Mixture of 1000 scan queries and 1M inserts
- Table with 10M tuples (~2GB)
DYNAMIC HYBRID WORKLOAD

- Row Store
- Column Store
- Hybrid Store

EXEUCUTION TIME (S)

QUERY SEQUENCE

Scan
Insert
Scan
Insert

30%

Row Store: Blue line
Column Store: Red line
Hybrid Store: Green line
“INCREMENTAL” LAYOUT ADAPTATION

**Immediate** vs. **Incremental**

**Scan-I** vs. **Scan-II**

**EXECUTION TIME (S)**

**QUERY SEQUENCE**

Carnegie Mellon Database Group
TAKEAWAYS

❖ Layout-agnostic execution engine
  • Enables real-time analytics with lower overheads

❖ Automatic layout reorganization module
  • Obviates the need for manual layout tuning
http://pelotondb.org
WE’RE ALWAYS LISTENING!
TOLL-FREE HOTLINE
1-844-88-CMUDB

END